Kartat ja kaaviot web-ohjelmoinnissa

- Tällä opintojaksolla esitellään ytimekkäästi vaihtoehtoja karttojen ja kaavioiden tuottamiseksi web-sovelluksissa. Tarkempiin yksityiskohtiin tulee perehtyä sekä harjoitustehtävien että mahdollisesti harjoitustyön avulla

Yleistä kartoista

- Kartta on paikkatiedon visuaalinen esitys ja sitä voidaan hyödyntää web-sovelluksissa havainnol-lis-tet-ta-es-sa
  - paikkojen tyyppiä ja sijaintia,
  - reittisuunnittelua,
  - alueisiin liittyviä ominaisuuksia kuten kiinteistötietoja, väestötiheyksiä, liikennedataa (LAM), jne.
  - jne.
- Karttadatan hyödyntämiseen web-sovelluksissa tarvitaan tyypillisesti
  - **karttadataa** eli paikkojen nimiä, osoitteita, tyyppejä (maasto, vesistö, tie, rakennus, ...) , korkeuksia jne. liitettynä sijainteihin.
  - **karttarajapinta** (API), jonka avulla karttadataa voidaan ainakin hakea ja mahdollisesti myös visualisoida useisiin käyttötarkoituksiin
  - **karttaohjelmakirjasto** helpottamaan karttarajapintojen käyttöä lopullisessa käyttötarkoituksissa
- Kaikkia edellisiä voi saada avoimena ilmaiseksi tai maksullisena palveluna joko yhdeltä toimijalta tai erikseen yhdistämällä erilaisia osia toisiinsa

Vaihtoehtoja karttojen tuottamiseksi

Google Maps Platform

Googlen karttoja EI vaadita käytettäväksi opintojakson harjoituksissa tai harjoitustöissä. Harjoitustyössä on kuitenkin oman harkinnan mukaan sallittua käyttää Googlen karttoja.

- Google Maps Platform tarjoaa datan, APIn ja ohjelmakirjaston yhdeltä toimijalta
- Erittäin laadukas karttadata, paljon ominaisuuksia, hyvin kuvattu API, Street View, maksullinen
- Opintojakson kannalta mielenkiintoisia palveluita ovat mm. Maps JavaScript API, Geocoding API ja Directions API

- Google Maps Platformin käyttöehdot ja laskutusmenettely muuttuivat kesällä 2018. Web-käyttöliittymässä palvelun hyödyntäminen edellyttää luottokorttitietojen luovuttamista. Vaikka pienillä käyttömäärillä opiskelutarkoituksissa laskutettavaa kertyy erittäin epätodennäköisesti, niin Googlen karttojen käyttöä ei edellytetä opintojaksolla. Harjoitustyössä voit toki oman harkintasi mukaan käyttää Googlen karttoja.

Leaflet-, Mapbox- ja OpenStreetMap-kokonaisuudet

- Leaflet-karttakirjasto hyödyntää MapBoxin säilömää ja OpenStreetMapillä ylläpidettäviä karttoja.
- Yhdistelmän käyttämiseksi on luotava MapBox APIn käyttöä varten tunnus ja "Access token". MapBox on tiettyyn käyttöön ja käyttöasteeseen asti ilmainen. Ilmaisuus riittää opiskelutarkoituksiin.
- Edellä lueteltujen kokonaisuuksien roolia on käsitelty mm. tässä keskusteluketjussa. Tiivistettynä ja mutkia oikoen:
  - Leaflet on karttaohjelmakirjasto ja karttarajapinta. Ei dataa
  - OpenStreetMap (OSM) on menetelmä, jolla luodaan, muokataan, säilötään ja tarjotaan karttanäkymiä (karttadataa)
  - MapBox on karttanäkymien säilö ja API, joka usein (mutta ei välttämättä) hyödyntää OSM-karttojen dataa. MapBox käyttää nykyään myös omassa APIssaan web-kehittäjille Leaflet-kirjastoa.

Openlayers

- Openlayers on täysin vapaa, erittäin laaja ja siksi tavanomaista korkeamman oppimiskynnyksen omaava karttakirjasto.
- pystyy lukemaan karttadataa useista lähteistä OSM, MapBox, Bing Maps, ... eri muodoissa m.. vektoridatanakin.

Openlayers vs. Leaflet

- **Leaflet**: Minimalistinen, nopeasti käyttöönotettava, helppo, kolmannen osapuolen plugineilla laajennettava => Sopii pieniin lyhyehkön elinkaaren sovelluksiin. "*ecosystem of 3rd party plugins*". Renderöinti DOM-elementteihin. Ei omaa karttadataa.
- **Openlayers**: erittäin laaja ydintoiminnallisuus ilman lisäosia, korkeahko oppimiskynnys. => Sopii myös laajoihin pitkän elinkaaren sovelluksiin. Renderöinti Canvas-elementiin. Ei omaa karttadataa.
- https://www.reddit.com/r/gis/comments/7x230v/openlayers_v4_vs_leaflet_advice/

Maanmittauslaitoksen karttatieto, kaspi

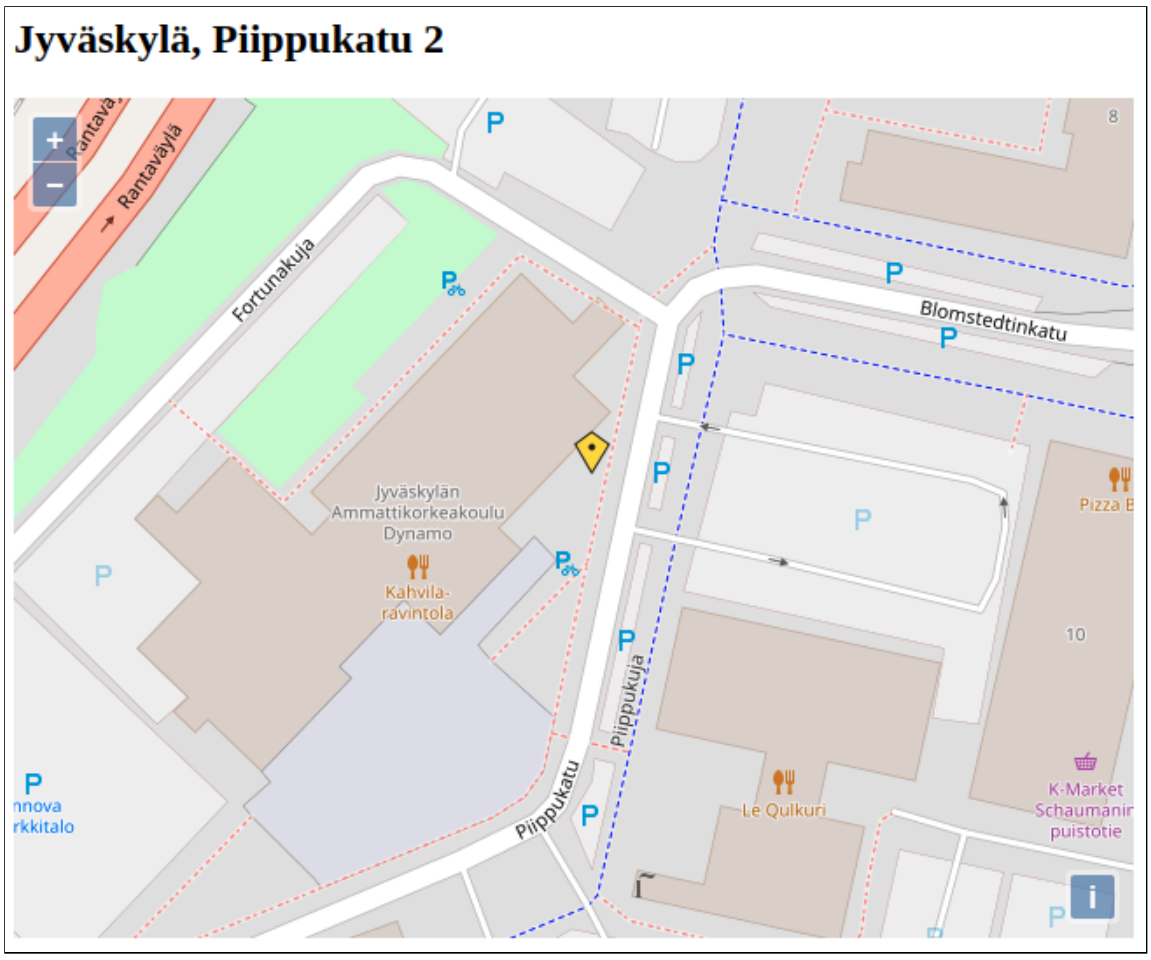
- Maanmittauslaitos (MML) tarjoaa avointa kartta- ja paikkatietoa rajapintapalveluna.
- Osaa MML:n karttoista on tuotu osaksi OSM-karttoja.
- Maanmittauslaitoksen karttadataa tarjoaa myös kartat.kaspi.fi-palvelu

Ks. myös hieman vanha opinnäytetyö aiheesta: Datan visualisointi kartalla

---

Karttaesimerkkejä

Openlayers OpenstreetMap-datalla



Lähdekoodi:

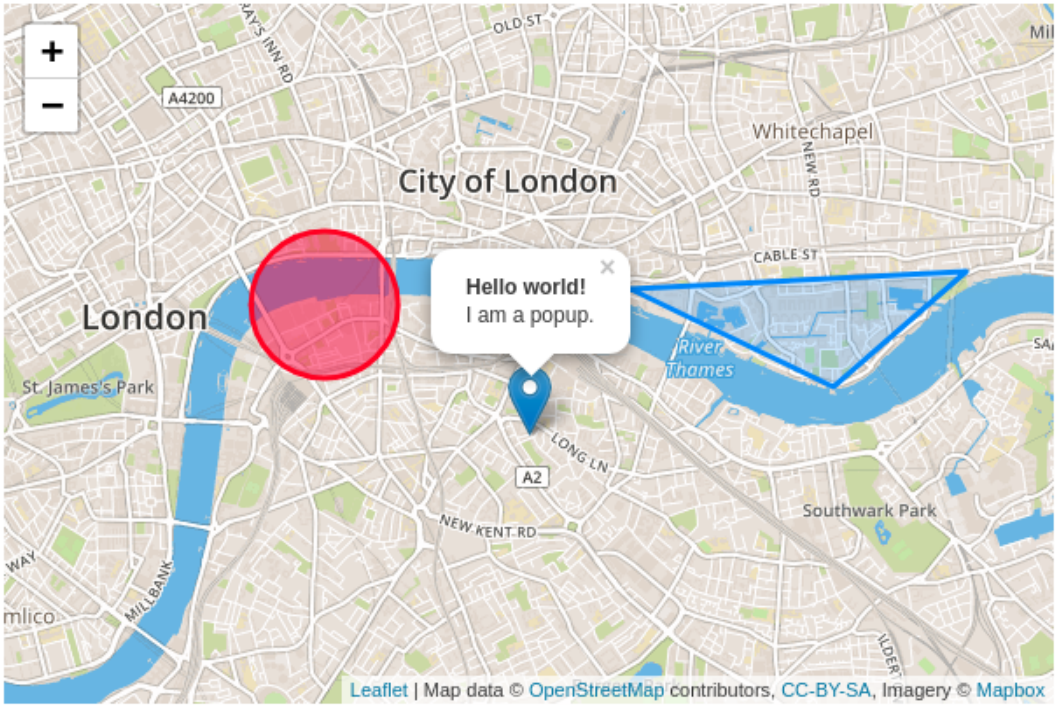
- Huom1: Markerin `marker.png` näkyminen vaatii sen tallentamista samaan kansioon ohjelman lähdekoodin kanssa.
- Huom2: Ei vaadi kirjautumista, API-avainta, lisenssointia... Täysin vapaa.

```
<!doctype html>
<html lang="en">
  <head>
    <link rel="stylesheet" href="https://cdn.rawgit.com/openlayers/openlayers.github.io/master/
    <style>
      .map {
        height: 480px;
```

```
        width:  640px;
      }
    </style>
    <script src="https://cdn.rawgit.com/openlayers/openlayers.github.io/master/en/v5.2.0/build/
    <title>OpenLayers example</title>
  </head>
  <body>
    <h2>Jyväskylä, Piippukatu 2</h2>
    <div id="map" class="map"></div>
    <script type="text/javascript">
      var map = new ol.Map({
        target: 'map',
        layers: [
          new ol.layer.Tile({
            source: new ol.source.OSM()
          })
        ],
        view: new ol.View({
          center: ol.proj.fromLonLat([25.7597309 ,62.24162229999999]),
          zoom: 18
        })
      });



var marker = new ol.Feature({
  geometry: new ol.geom.Point(
    ol.proj.fromLonLat([25.7597309 ,62.24162229999999])
  ),  // Coordinates of Piippukatu 2
});

  var iconStyle = new ol.style.Style({
    image: new ol.style.Icon(/** @type {olx.style.IconOptions} */ ({
      anchor: [0, 50],
      anchorXUnits: 'pixels',
      anchorYUnits: 'pixels',
      opacity: 0.90,
      src: 'marker.png'
    }))
  });

  marker.setStyle(iconStyle);


var vectorSource = new ol.source.Vector({
  features: [marker],
});
var markerVectorLayer = new ol.layer.Vector({
  source: vectorSource,
});
map.addLayer(markerVectorLayer);




    </script>
  </body>
</html>
```

Leaflet/MapBox-kartta OSM-datalla

Eka Leaflet-kartta



Lähdekoodi

- Huom: Vaatii MapBox APIn käyttöä varten tunnuksen ja "Access tokenin".

```
aas


<!DOCTYPE html>
<html>

   <head>
      <title>Eka HTML-dokumentti</title>
      <meta charset="UTF-8">

<link rel="stylesheet" href="https://unpkg.com/leaflet@1.3.3/dist/leaflet.css"
   integrity="sha512-Rksm5RenBEKSKFjgI3a41vrjkw4EVPlJ3+OiI65vTjIdo9brlAacEuKOiQ5OFh7cOI1bkDwLqd
   crossorigin=""/>

 <!-- Make sure you put this AFTER Leaflet's CSS -->
 <script src="https://unpkg.com/leaflet@1.3.3/dist/leaflet.js"
   integrity="sha512-tAGcCfR4Sc5ZP5ZoVz0quoZDYX5aCtEm/eu1KhSLj2c9eFrylXZknQYmxUssFaVJKvvc0dJQi
   crossorigin=""></script>


<style type="text/css">
   #mapid { height: 400px; width: 600px;}
</style>

   </head>

   <body>
      <h1>Eka Leaflet-kartta</h1>

<div id="mapid"></div>




<script>
var mymap = L.map('mapid').setView([51.505, -0.09], 13);
```

```
L.tileLayer('https://api.tiles.mapbox.com/v4/{id}/{z}/{x}/{y}.png?access_token=pk.eyJ1IjoieXVya
    attribution: 'Map data &copy; <a href="https://www.openstreetmap.org/">OpenStreetMap</a> co
    maxZoom: 18,
    id: 'mapbox.streets'
}).addTo(mymap);


var marker = L.marker([51.5, -0.09]).addTo(mymap);

var circle = L.circle([51.508, -0.11], {
    color: 'red',
    fillColor: '#f03',
    fillOpacity: 0.5,
    radius: 500
}).addTo(mymap);

var polygon = L.polygon([
    [51.509, -0.08],
    [51.503, -0.06],
    [51.51, -0.047]
]).addTo(mymap);


marker.bindPopup("<b>Hello world!</b><br>I am a popup.").openPopup();
circle.bindPopup("I am a circle.");
polygon.bindPopup("I am a polygon.");



/* Eka esimerkki klikkauksesta*/

/*
function onMapClick(e) {
    alert("You clicked the map at " + e.latlng);
}

mymap.on('click', onMapClick);
*/


var popup = L.popup();

function onMapClick(e) {
    popup
        .setLatLng(e.latlng)
        .setContent("You clicked the map at " + e.latlng.toString())
        .openOn(mymap);
}

mymap.on('click', onMapClick);




</script>


   </body>

</html>
```

1. API-avain

- Google Maps JavaScript APIn käyttö **edellyttää** autentikointia ja tämä todennetaan käyttämällä API-avainta, jonka sovelluskehittelijä saa Googlen API Consolen kautta.
- Ohjeet API-avaimen saantiin löytyvät täältä: Get a API key.

2. Sovelluksen ohjelmointi

- Karttasovellusten ohjelmointi **kannattaa** aloittaa Getting Started-sivuston esimerkkeihin tutustumisella
- Sisältää erittäin paljon havainnollisia esimerkkejä kartan ohjelmointiin.

Google - Käyttäjän laitteen/tietokoneen sijainti

- Suurin osa selaimista voi suorittaa käyttäjän laitteen paikannuksen JavaScript-ohjelmoinnein
- `GeoLocation`-ohjelmointirajapinta kuvataan W3C:n sivuilla Geolocation API Specification
- Sijainti lasketaan ensisijaisesti GPS-signaalista, jos sellainen on saatavilla. Muussa tapauksessa sijanti yritetään saada selville IP-osoitteen tai esim. mobiilitukiasemien kautta. Yleisesti sijainti näytetään esim. Googlen kartan avulla.

> Huom! getCurrentPosition()- ja/tai watchPosition()-funktioiden käyttö sallitaan ainoastaan **https**-yhteyden kautta. Sovelluksia voi testata myös localhost-yhteyden avulla.
>
> Loppukäyttäjä hyväksyy lopulta aina paikantamisen sallimisen!

Paikannus

Tarkista onko toiminto tuettu käyttämässäsi selaimessa, joko tutkimalla globaalia navigator-objektia.

```
if (navigator.geolocation) {
  navigator.geolocation.getCurrentPosition(showPosition, handleError);
} else {
  $("#location").text("No native support for Geolocation API");
}
```

Näytetään löytynyt sijainti

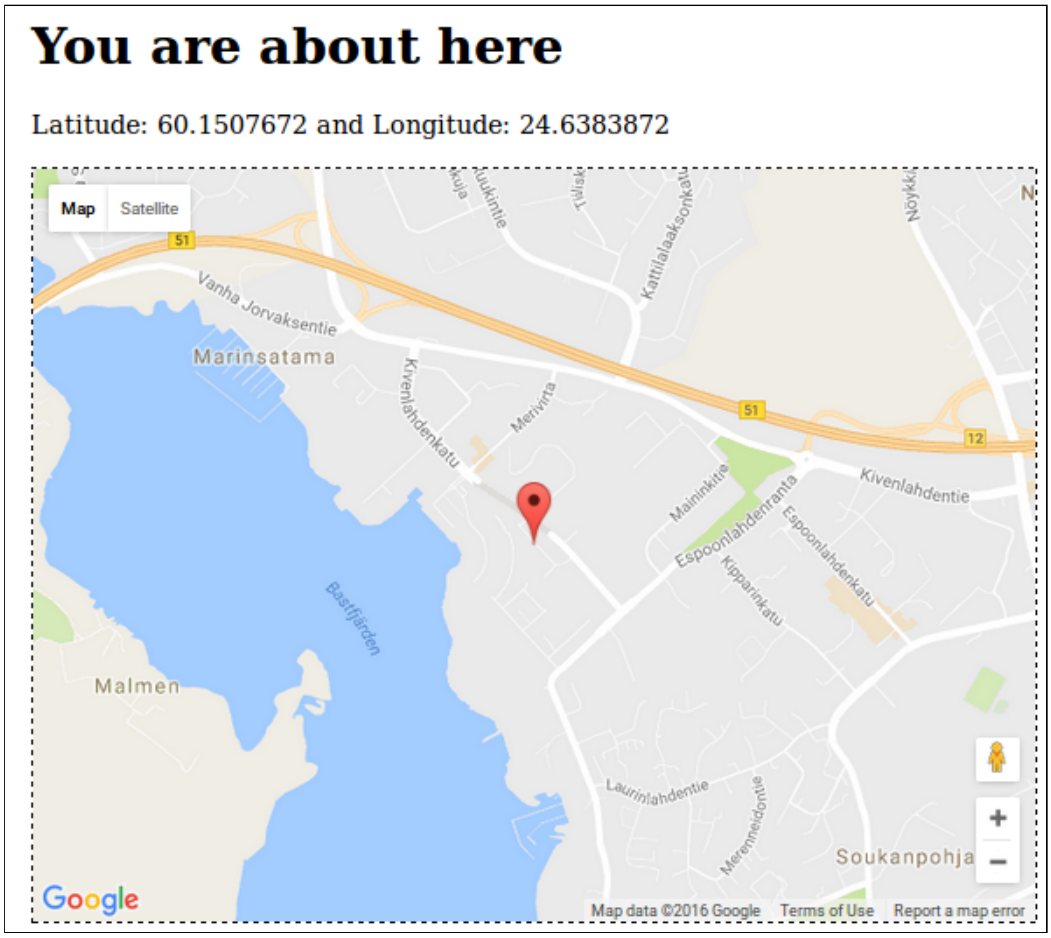
```
function showPosition(position) {
  var latitude = position.coords.latitude;
  var longitude = position.coords.longitude;
  $("#location").text("Latitude: "+ position.coords.latitude + " and Longitude: " + position.co
}
```

tai käsitellään mahdollinen virhetilanne.

```
function handleError(error) {
  if (error.code == error.PERMISSION_DENIED) {
    $("#location").text("Permission denied by the user.");
  }
  else if (error.code == error.POSITION_UNAVAILABLE) {
    $("#location").text("Position is unavailable");
  }
  else if (error.code == error.TIMEOUT) {
    $("#location").text("Position can't be found - timeout.");
  }
  else {
    $("#location").text("Something wierd happend!");
  }
}
```

Google - Esimerkki 0701 - Käyttäjän sijainti kartalle

Toteutetaan pieni sovellus, joka näyttää käyttäjän sijainnin kartalla.



API-key

Luo oma projekti Google API Consolen kautta. Luodaan projektiin oma Google Maps JavaScript API-avain ja rajoitetaan sen toimivuus student.labranet.jamk.fi/~opnro ja

http://localhost-toimivuusalueelle. Näin ollen kukaan muu ei pysty käyttämään luotua avainta.

Accept request from these HTTP referrers -kohtaan määritellään: student.labranet.jamk.fi/~opinro/* sekä http://localhost/*

HTML

Muodostetaan web-sivu, jossa käyttäjän sijaintiin liittyvät tiedot näytetään sekä tekstikappaleessa että kartalla.

```
<div>
    <h1>You are about here</h1>
    <p id="location">Location:</p>
</div>
<div id="map"></div>
```

HTML-osuuteen liitetään myös tarvittavien kirjastojen lataukset SCRIPT-elementteinä. Google Maps API kutsuu getPosition-funktiota, kun kartta on valmiina käytettäväksi. Toteutetaan getPosition-funktio omaan showLocation.js-tiedostoon.

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.1.0/jquery.min.js"></script>
<script src="js/showLocation.js"></script>
<script src="https://maps.googleapis.com/maps/api/js?key=yourapikeyhere&callback=getPosition" a
```

JavaScript-osuus

Selvitetään `getPosition()` -funktiossa käyttäjän sijainti käyttämällä `navigator.geolocation.getCurrentPosition()` -funktiota. Onnistuneessa tilanteessa kutsutaan omaa `showPosition()` -funktiota näyttämään käyttäjän sijainti kartalla.

```
// get position
function getPosition(position) {
    // try HTML5 geolocation
    if (navigator.geolocation) {
        // get current position, you need to run this on localhost or https to get it working
        navigator.geolocation.getCurrentPosition(showPosition,handleError);
    } else {
        // your browser doesn't support Geolocation
        $("#location").text("browser doesn't support Geolocation");
    }



// show position
function showPosition(position) {
    var pos = {
      lat: position.coords.latitude,
      lng: position.coords.longitude
    };
    // show location in HTML
    $("#location").text("Latitude: "+ pos.lat + " and Longitude: " + pos.lng);
    // point map to location and zoom a little
    var map = new google.maps.Map(document.getElementById('map'), {
        center: pos,
        zoom: 14
    });
```

```
    // add one marker
    var marker = new google.maps.Marker({position:pos, map:map, title:"You are about here!"});
}
```

Käsitellään myös mahdollinen virhetilanne sijainnin löytämisen suhteen.

```
// show error
function handleError(error) {
  if (error.code == error.PERMISSION_DENIED) {
    $("#location").text("Permission denied by the user.");
  }
  else if (error.code == error.POSITION_UNAVAILABLE) {
    $("#location").text("Position is unavailable");
  }
  else if (error.code == error.TIMEOUT) {
    $("#location").text("Position can't be found - timeout.");
  }
  else {
    $("#location").text("Unknow error has happend");
  }
}
```

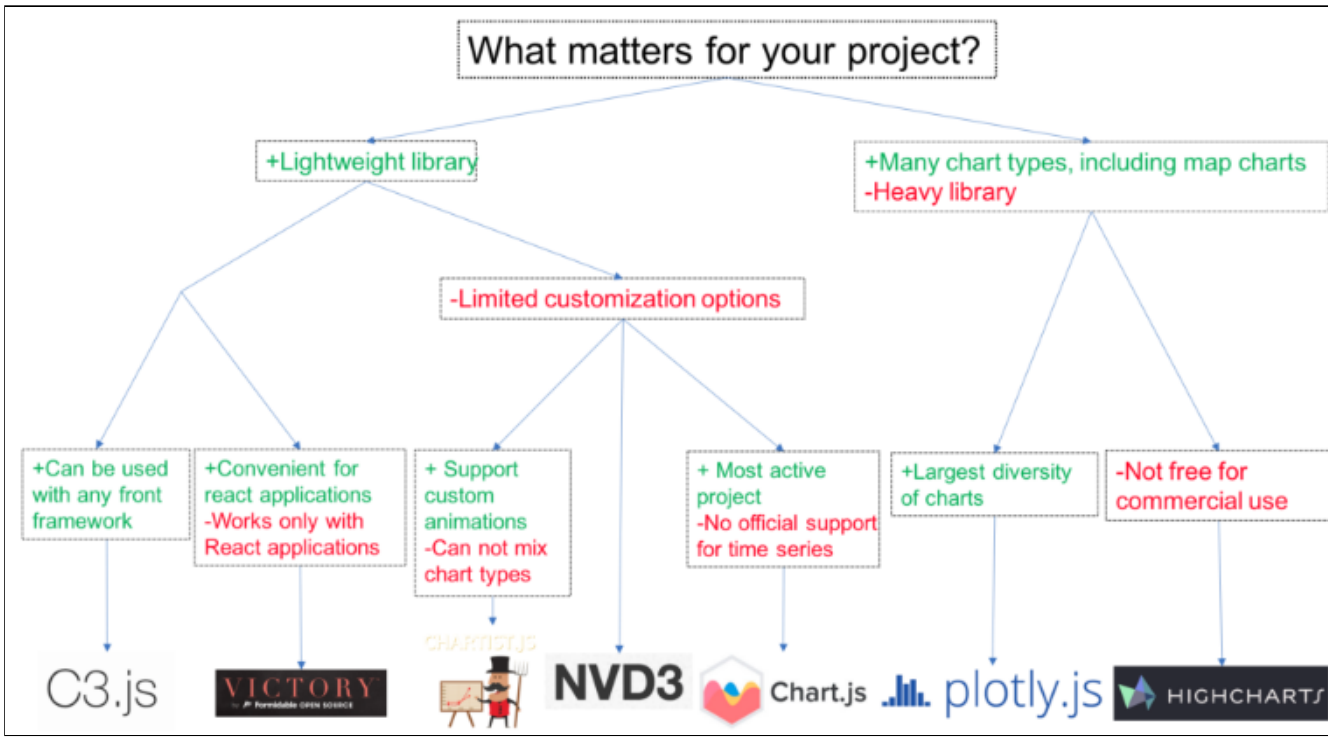
JavaScript-kaaviokirjastot

- Kaavioiden luominen Canvas-elementtiin on helppoa, mutta ammattimaisen jäljen tuottaminen on työlästä
- Tarjolla lukuisia laadukkaita ja ilmaisia JavaScript-kirjastoja numeerisen datan visualisointiin kaavioiden avulla

Merkittäviä JavaScript-kaaviokirjastoja

- chart.js - kevyt, helppo, paljon käytetty, aktiivisessa kehityksessä, avoin lähdekoodi, piirtäminen canvasiin
- Highcharts - ilmainen henkilökohtaiseen käyttöön
- Plotly.js - erittäin laaja, kaikille ilmainen
- Chartist - svg, community
- Flot - perinteinen yksi vanhismmista jQuerya käyttävä. Täsyin vapaa
- Google Charts - taattua Google-laatua
- Muitakin on...

Eräs valintakartta

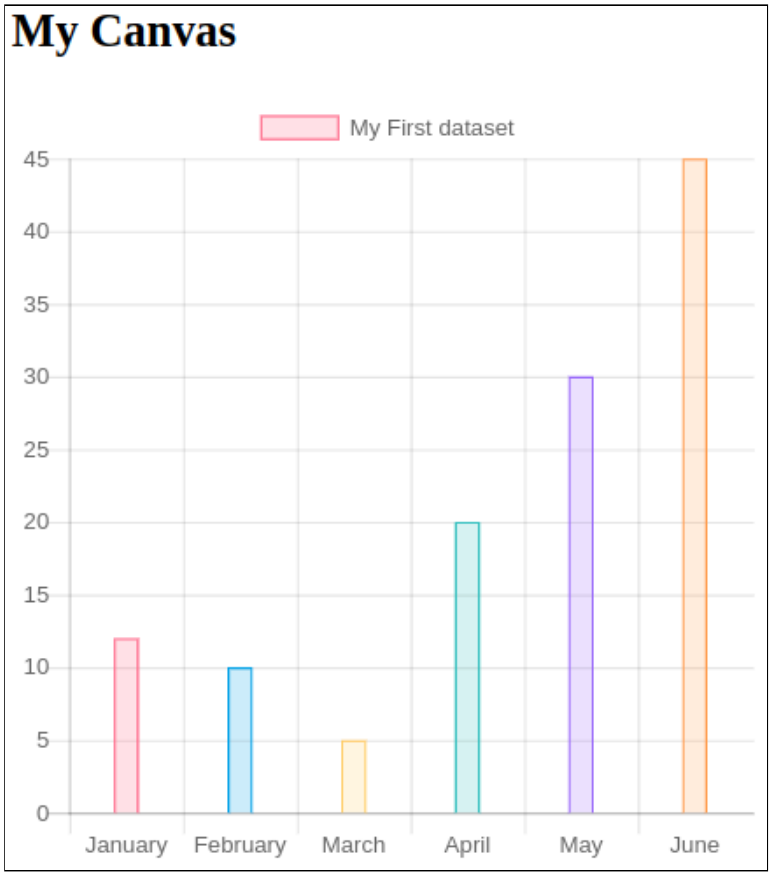


Lähde: https://blog.sicara.com/compare-best-javascript-chart-libraries-2017-89fbe8cb112d

Vertailuja

- https://blog.sicara.com/compare-best-javascript-chart-libraries-2017-89fbe8cb112d
- https://www.sitepoint.com/15-best-javascript-charting-libraries/
- https://hackernoon.com/9-best-javascript-charting-libraries-46e7f4dc34e6

chart.js-esimerkki



Lähdekoodi

```
<html>
<title>Chart.js-esimerkki</title>
<body>

<h2>My Canvas</h2>
<canvas id="myChart" width="400" height="400"></canvas>

<script src="https://cdnjs.cloudflare.com/ajax/libs/Chart.js/2.7.2/Chart.js"></script>
<script>
var ctx = document.getElementById("myChart").getContext('2d');
var chart = new Chart(ctx, {
    // The type of chart we want to create
//    type: 'horizontalBar',
    type: 'bar',

    // The data for our dataset
    data: {
        labels: ["January", "February", "March", "April", "May", "June"],
        datasets: [{
            label: "My First dataset",
            data: [12, 10, 5, 20, 30, 45],
            backgroundColor: [
                'rgba(255, 99, 132, 0.2)',
                'rgba(54, 162, 235, 0.2)',
                'rgba(255, 206, 86, 0.2)',
                'rgba(75, 192, 192, 0.2)',
                'rgba(153, 102, 255, 0.2)',
                'rgba(255, 159, 64, 0.2)'
            ],
            borderColor: [
                'rgba(255,99,132,1)',
                'rgba(54, 162, 235, 1)',
                'rgba(255, 206, 86, 1)',
                'rgba(75, 192, 192, 1)',
                'rgba(153, 102, 255, 1)',
                'rgba(255, 159, 64, 1)'
```

```
            ],
            borderWidth: 1
        }]
    },

    // Configuration options go here
    options: {
        responsive: false,
        scales: {
            yAxes: [{
                barThickness: 53,
                ticks: {
                    beginAtZero:true
                }
            }],
            xAxes: [{
                barThickness: 13
            }]
        }
   }
});
</script>



</body></html>
```

Lisätietoa

Google-esimerkkejä Huomaa, https-linkki, että geolocation toimisi suoraan. API-keyt vanhoja, saattaa lakata toimimasta koska tahansa.

Jätetty tarkoituksella tyhjäksi.